UNITED STATES PATENT APPLICATION

FOR

SYSTEMS AND METHODS FOR FREEFORM ANNOTATIONS

Inventors

Laurent Denoue
Gene Golovchinsky

CERTIFICATE OF MAILING BY "EXPRESS MAIL" UNDER 37 C.F.R. §1.10

"Express Mail" mailing label number: EV 386447581 US
Date of Mailing: Feb. 10, 2004

I hereby certify that this correspondence is being deposited with the United States Postal Service, utilizing the "Express Mail Post Office to Addressee" service addressed to: **Mail Stop PATENT APPLICATION, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450** and mailed on the above Date of Mailing with the above "Express Mail" mailing label number.

Olivia M. Jomoc (Signature)
Olivia M. Jomoc
Signature Date: Feb. 10, 2004

SYSTEMS AND METHODS FOR FREEFORM ANNOTATIONS

Inventors

Laurent Denoue
Gene Golovchinsky

COPYRIGHT NOTICE

FIELD OF THE DISCLOSURE

[0002] The present disclosure relates to the input, processing and display of freeform annotations.

BACKGROUND

[0003] Freeform annotations are ink-based strokes, gestures, or handwritings of any shape that a user can input to a computing device using an associated input device, which can be but is not limited to, a mouse, a keyboard, and a stylus. Here a computing device can be but is not limited to, a PDA, a Tablet PC, a Pocket PC, a cell phone, an electronic messaging device, a Java-enabled device, a laptop or desktop PC, a workstation, and a mainframe computer. The processing unit inside the computing device can be a CPU, an embedded CPU, or a Micro Control Unit (MCU). Freeform annotation systems have been pursued since, unlike text-based annotation systems that do not support natural annotation as on paper, they provide a natural way to annotate digital documents, which is especially important for portable computing devices such as: PDA, Pocket PC or cell phone. A digital document can be stored either in memory or on a persistent storage (e.g., hard drive) associated with the computing device, and it can include at least one of: a text file, an image, a figure, a drawing, a graph, a picture, and a video clip.

[0004] Previous approaches to freeform annotation systems suffer from two major drawbacks: First, they often limit the kind of annotations that can be recognized and put restrictions on how the annotations can input in order to be recognized correctly, making

the annotations not truly "freeform". For example, links between portions of a displayed digital document are often not recognized; Secondly, they often rely on heuristic techniques to automatically group individual annotations into various categories. Due to their heuristic nature, these techniques do not work all the time.

[0005] The situation is further complicated when the digital document is viewed on different display devices or when users select different font sizes, requiring the annotations on the document to be repositioned or resized according to the display properties of a specific display device in order to be rendered properly. Here a display device can be a screen, which can be a LCD or a CRT monitor having a display area of certain width and height measured in terms of number of pixels; A display property can be associated with one of: a text font, a font size, a text color, a display width, height, and resolution of the display device. Text-based annotation systems can provide an obvious place or a special text box to input annotations, making rendering of the annotations a straightforward task. Freeform annotation systems, however, require that the annotations to be linked to a portion of the digital document for proper rendering. In addition, certain freeform annotations, such as comments, have to be sized down from handwriting sizes in order to fit within the often limited blank area available on a display device, such as the screen of a PDA.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] **Figure 1** is an illustration of an exemplary freeform annotation system in accordance to one embodiment of the invention.

[0007] **Figure 2** is an illustration of exemplary freeform annotations in accordance to one embodiment of the invention.

[0008] **Figure 3** is an illustration of an exemplary code of an algorithm implementing a state machine in accordance to one embodiment of the invention.

[0009] **Figure 4** is an illustration of an exemplary misplaced freeform annotation in accordance to one embodiment of the invention.

[0010] **Figure 5 (a)-(d)** are illustrations of methods marking the completion of a comment in accordance to one embodiment of the invention.

[0011] **Figure 6** is an illustration of the rendering of a comment in accordance to one embodiment of the invention.

[0012] **Figure 7** is an illustration of resizing and repositioning of exemplary freeform annotations in accordance to one embodiment of the invention.

DETAILED DESCRIPTION

[0013] The invention is illustrated by way of example and not by way of limitation in the figures of the accompanying drawings in which like references indicate similar elements. It should be noted that references to "an" or "one" embodiment in this disclosure are not necessarily to the same embodiment, and such references mean at least one.

[0014] Systems and methods in accordance with the present invention provide freeform annotation techniques designed to make freeform annotations on a digital document more flexible, reliable and presentable on a screen of a computing device. They allow users to markup the document, add comments, and link between annotations, while maintaining the freeform nature of these annotations without constraining the kinds of annotations allowed. These freeform annotations are then unequivocally recognized via a state machine into three groups of annotation marks: anchor, comment, and link, each of which is associated with either a portion of the digital document or with another annotation mark. In addition, these annotation marks can be repositioned and resized to fit within the blank area available on the screen based on the display properties of the screen, thus supporting the natural flow of annotating the document in a freeform manner. The software used in the system can be implemented in one or more of the following programming languages: C, C++, Visual Basic, Java and Java Applet.

[0015] **Figure 1** is an illustration of an exemplary freeform annotation system in an embodiment of the present invention. Although this diagram depicts objects/processes as logically separate, such depiction is merely for illustrative purposes. It will be apparent to those skilled in the art that the objects/processes portrayed in this figure can be arbitrarily combined or divided into separate software, firmware and/or hardware components. Furthermore, it will also be apparent to those skilled in the art that such objects/processes, regardless of how they are combined or divided, can execute on the same computing device or can be distributed among different computing devices connected by one or more networks or other suitable communication means.

[0016] Within the exemplary freeform annotation system **100** in **Figure 1**, a digital document **101** can be stored in a storage component **111** either in memory or on a persistent storage in a computing device. A display component **102** reads a portion of the digital document through a document reader **103** and display it on its screen **104** based on the display properties **105** of the screen. A user can mark and comment on the displayed

portion of the digital document using freeform annotations via one or more input devices **106**. A processing component **107** then accepts and interprets these freeform annotations using a state machine **108** and generates three categories of annotation marks. The state machine is completely deterministic, i.e., it can always unequivocally recognize and characterize those freeform annotations. These generated annotation marks are associated with the corresponding portions of displayed document, and can be selected, moved, modified and deleted. They can be stored in various kinds of data structures **109**, either in memory or in persistent storage. Furthermore, they can be repositioned, resized and rendered for display by a rendering component **110** based on the display properties of the screen.

[0017] In one embodiment, the document reader **103** needs to have access to the bounding boxes of all words and images in the portion of the digital document under display. An application implemented as a Java applet is developed that accepts an URL and the width of the screen for display, e.g. 800 pixels for a desktop, 240 pixels for a Pocket PC screen, It returns an image of a Web page rendered at the specified width, and also a file containing the bounding boxes of all elements on that page. By changing the width, the application can easily generate layouts that are adapted to different screen sizes.

[0018] In one embodiment, the Microsoft Internet Explorer control can be used by the application to render the Web page by capturing its rendition, and the Document Object Model can also be accessed to read the bounding box information. This information is then used by the application to display the rendered image, making it suitable for annotating Web pages on any Java-enabled device like most laptops and PDAs.

[0019] In one embodiment, the processing component **107** recognizes and characterizes annotation marks into three groups:

- Anchors, which include: highlights, underlines, margin bars, circled words, circled areas, brackets, etc. An anchor can identify a part of the portion of the digital document under display to which a comment might eventually be attached;
- Comments, which are usually handwritten notes. Comments are often written in a margin on the document, but are more generally written on a blank area of the document where there is space to write;
- Links, which include connectors and arrows. A link can link between anchors, anchors and comments, or multiple comments.

Figure 2 shows a portion of a Web page displayed with anchors **201** and **202**, a link **203**,

and a comment **204**, all can be marked in any color.

[0020] In one embodiment, the state machine **108** in the processing component is designed to recognize the natural flow of freeform annotations on a digital document under display, creating anchors, adding comments, and creating links. **Figure 3** is an exemplary source code illustrating an algorithm implementing the state machine in accordance to one embodiment of the invention. Although this figure depicts functional steps in a particular order for purposes of illustration, the process is not limited to any particular order or arrangement of steps. One skilled in the art will appreciate that the various steps portrayed in this figure could be omitted, rearranged, combined and/or adapted in various ways.

[0021] Referring to **Figure 3**, the state is initially set to FIRST_STROKE. In this state, the state machine always tries to interpret a stroke as an anchor or a link. When a stroke has successfully been recognized as an anchor or a link between two existing anchors, the state goes to ANCHORING.

[0022] While in the ANCHORING state, the state machine first tries to interpret a stroke as an anchor or a link. If the stroke is not recognized as an anchor or a link (e.g. the stroke starts on an blank area of the document), then the state switches to COMMENTING and the stroke is added to a new comment (where a comment is a collection of strokes).

[0023] While in the COMMENTING state, the state machine tries to interpret if the stroke indicates exiting the commenting mode. If not, the stroke gets added to the same comment (i.e. same collection of strokes). However, if the system has detected that the stroke indicates exiting the commenting mode (e.g. clicking on a button, a special gesture, and other indications as discussed later), then the state can switch back to FIRST_STROKE or FREEFORM_PASTING, depending on user preferences.

[0024] In FREEFORM_PASTING state, users can input one more stroke that specify where and how the newly created collection of strokes (i.e., current comment) should be rendered. After that stroke is input, the state goes back to FIRST_STROKE.

[0025] When a stroke is finished, the state machine tries to interpret it as follows (**401-404** shown in **Figure 4** correspond to **201-204** in **Figure 2,** respectively):

- If the stroke intersects a word or an image, then it is interpreted as an anchor **401** or **402**. The bounding boxes of the words that have been identified by this anchor are displayed, providing a useful feedback to the user.
- If the stroke does not intersect any word or image, it is interpreted as a comment **404**.

The state machine switches to the COMMENTING state and the strokes are added to the comment.

- If the stroke **405** does not overlap with any word or image while in the ANCHORING state, the system redraws it in a distinctive color, to indicate that it did not recognize it, as shown in **Figure 4**. Unrecognized marks disappear after a little while.
- A link **403** is detected if the origin and the destination of the stroke overlap an existing anchor or comment.

[0026] The algorithm above can make sure that the state machine is always able to unequivocally group strokes properly by adopting the following rules:

- Strokes making a comment are assembled into a single comment (collection of strokes) and labeled as such;
- A comment is always linked to an anchor or a link (i.e., the stroke that was created right before the comment started);
- An anchor is always linked to a specific part of the displayed portion of the digital document; and
- A link always identifies two existing anchors.

[0027] In one embodiment, several approaches can be adopted by the state machine to accurately recognize when the user has finished commenting, these approaches can include:

- Clicking on a pre-defined area of the screen, for example, a button **501** that is displayed next to the first stroke of the comment, as shown in **Figure 5 (a).**
- Selecting the commenting strokes with a circular gesture **502** that includes most of the strokes input since the comment was started, as shown in **Figure 5 (b).**
- Selecting the commenting strokes with a strike-through gesture **503**, as shown in **Figure 5 (c).**
- Drawing a stroke **504** that bounces off at the bottom of the screen, as shown in **Figure 5 (d)**.
- Using a special gesture that is preferably rare in common handwriting, so that the state machine does not exit the COMMENTING state inappropriately.
- Drawing a vertical stroke on the left or right of the commenting strokes.
- Etc.

[0028] After one of these approaches is taken, the state machine can either directly

switch back to ANCHORING state or, depending on user preferences or the method chosen to end the COMMENTING state, the state machine can also go to the FREEFORM_PASTING state. In the later case, users can input another stroke to specify how and where the comment should be rendered in the document. As shown in **Figure 6**, the user specifies such a stroke **601** and the state machine renders the comment along the stroke. Being able to specify where and how the comment is pasted is useful on PDAs where users are forced to write big.

[0029] In one embodiment, areas are reserved on the screen for the user to input freeform annotations. It often happens that the screen displaying a portion of the digital document might be so crowded on that it might be difficult for the user to find a blank area from where he/she can start commenting. In one approach, left and right margins are created automatically so that users can always (and predictably) find a blank area. This feature is very important when the page is accessed from a PDA because the image might not be displayed at original captured scaled, but zoomed out a little bit to prevent too much panning. In another approach, blank areas are visually made salient from the bounding boxes of the text and images, helping users know where they can ink to switch to commenting. Yet in another approach, an area (e.g. a button) is defined on the screen where users can click to specifically start commenting.

[0030] In one embodiment, the state machine is capable of identifying the following anchor types by recognizing:

- Circled words/areas: end point of stroke near start of stroke.
- Highlights/underlines: the width of bounding box is at least three times larger than its height.
- Margin bars: height is at least three times larger than its width.

If a stroke does not fall within these types, the rendering component **110** indicates the mis-recognition by redrawing the stroke in a distinctive color and then erases it. When the stroke is recognized as one type of the anchor, the rendering component then associates some document content to that anchor as follows:

- Highlights/underlines: the words that are closest to the stroke.
- Margin bars: the word range, where the first word being the word closest to the top of the stroke, the last word being the last word on the line aligned vertically with the bottom of the stroke.
- Circles words/areas: the word that is closest to the center of the bounding box of the stroke.

[0031] In one embodiment, annotation marks are stored in the following types of data structures **109** once they are recognized, either in memory or on a persistent storage:

- Anchor: author, time, URL, anchor stroke, text selected by the anchor stroke, height of the text selected by the stroke.
- Comment: author, time, URL, comment stroke, anchor.
- Link: author, time, URL, link stroke, first anchor or comment, second anchor or comment (optional).

Here, a stroke can represent a collection of ink segments. When the user starts entering a comment, all strokes that are input are grouped together into the same stroke. The point coordinates are stored as (x, y) coordinates, and a break is represented by a (-1,-1).

[0032] In one embodiment, users might want to see the links between a specific comment and its associated anchor, or the anchors linked by a connector. At any time, users can tap and hold (mouse over with a mouse) over a stroke: the rendering component automatically highlights the corresponding objects. Since annotations can be created by different authors, re-coloring or other highlighting procedures can be used by the rendering component to show all annotations made by a specific person or at a particular time/data.

[0033] In one embodiment, annotation marks can be selected and modified in one of the following steps:

- Adding text to an existing comment: tapping, holding, and lifting stylus (or mouse up) on top of an existing comment allow users to add more strokes to this comment. On a PDA, comment strokes are zoomed in back to their original size to allow comfortable handwriting.
- Moving/delete anchors, links, comments: tapping, holding and dragging allow users to move/delete anchors. When an anchor is deleted, its associated comment and/or connector is also removed.

[0034] Once the processing component understands anchors, links and comments, the rendering component **110** is able to correctly reposition and resize annotations when the document layout changes, as it often happens. Being able to correctly categorize marks into these three groups becomes even more important on small screens such as PDAs: users need to write much bigger and the rendering component might want to automatically scale down the comment once finished. It is also important when users start sharing their annotations because they want to make sure that the system will correctly reposition their annotations when the same document is displayed on their colleagues'

display components having different display properties such as: screen size, font size, etc.

[0035] In one embodiment, the rendering component in the processing component adopts an algorithm to resize and reposition anchors according to their types as follows (**701-704** shown in **Figure 7** correspond to **201-204** in **Figure 2,** respectively):

- Circled area/word: the rendering component locates the bounding box of the word to which the anchor **201** in **Figure 2** was attached, redraws the original stroke, possibly scaled according to the size of the new bounding box of the word in the document, and reposition it on the screen, shown as **701** in **Figure 7**. This is important if the user has changed the font size.
- Margin bar: the rendering component locates the first and last words of the word range in the document and computes the height of the bounding box. The margin bar is scaled vertically.
- Underline/highlight: the rendering component retrieves the bounding box of the word range. If the words still appear on the same line, the rendering component scales the stroke horizontally. If the words appear on different lines, the rendering component splits the mark and scale the new segment to the length of the remaining words. For example, the underline **202** in **Figure 2** is split into **702a** and **702b** as shown in **Figure 7**.

[0036] If comments **204** in **Figure 2** were on the right of their associated anchor, the rendering component displays them on a blank area of the document located on the right of the page, shown as **704** in **Figure 7**. The rendering component does the opposite if the comment was located on the left of their associated anchor. If they were above their associated anchor, the rendering component displays them above in the blank area, and vice versa if they were below their associated anchor. To minimize overlap, the rendering component repositions comments vertically so that they don't overlap previously repositioned comments. Comments are automatically scaled down when displayed on a smaller screen.

[0037] In one embodiment, a link **203** in **Figure 2** is replaced by a straight line **703** connecting the two anchors, as shown in **Figure 7**. The start and end points of the line are computed once the anchors have been repositioned: they are set to the center of the bounding box of the anchor. Finer grained repositioning is possible by storing the original location of the start/end points of the links within the original anchors' bounding boxes.

[0038] When freeform pasting is not used, the rendering component can automatically scale down the commenting strokes before rendering, starting at the top/left corner of the

bounding box of the original comment. This is especially useful on small screens where users are forced to write big.

[0039] One embodiment may be implemented using a conventional general purpose or a specialized digital computer or microprocessor(s) programmed according to the teachings of the present disclosure, as will be apparent to those skilled in the computer art. Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present disclosure, as will be apparent to those skilled in the software art. The invention may also be implemented by the preparation of integrated circuits or by interconnecting an appropriate network of conventional component circuits, as will be readily apparent to those skilled in the art.

[0040] One embodiment includes a computer program product which is a storage medium (media) having instructions stored thereon/in which can be used to program a computer to perform any of the features presented herein. The storage medium can include, but is not limited to, any type of disk including floppy disks, optical discs, DVD, CD-ROMs, micro drive, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, DRAMs, VRAMs, flash memory devices, magnetic or optical cards, nanosystems (including molecular memory ICs), or any type of media or device suitable for storing instructions and/or data.

[0041] Stored on any one of the computer readable medium (media), the present invention includes software for controlling both the hardware of the general purpose/specialized computer or microprocessor, and for enabling the computer or microprocessor to interact with a human user or other mechanism utilizing the results of the present invention. Such software may include, but is not limited to, device drivers, operating systems, execution environments/containers, and applications.

[0042] The foregoing description of the preferred embodiments of the present invention has been provided for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many modifications and variations will be apparent to the practitioner skilled in the art. Particularly, while the concept "freeform annotation" is used for in the embodiments of the systems and methods described above, it will be evident that such concept can be interchangeably used with equivalent concepts such as, stroke, gesture, handwriting, mark, and other suitable concepts; While the concept "annotation mark" is used for in the embodiments of the systems and methods described above, it will be evident that such concept can be interchangeably used with equivalent concepts such as, annotation, and other suitable

concepts. Embodiments were chosen and described in order to best describe the principles of the invention and its practical application, thereby enabling others skilled in the art to understand the invention, the various embodiments and with various modifications that are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the following claims and their equivalents.